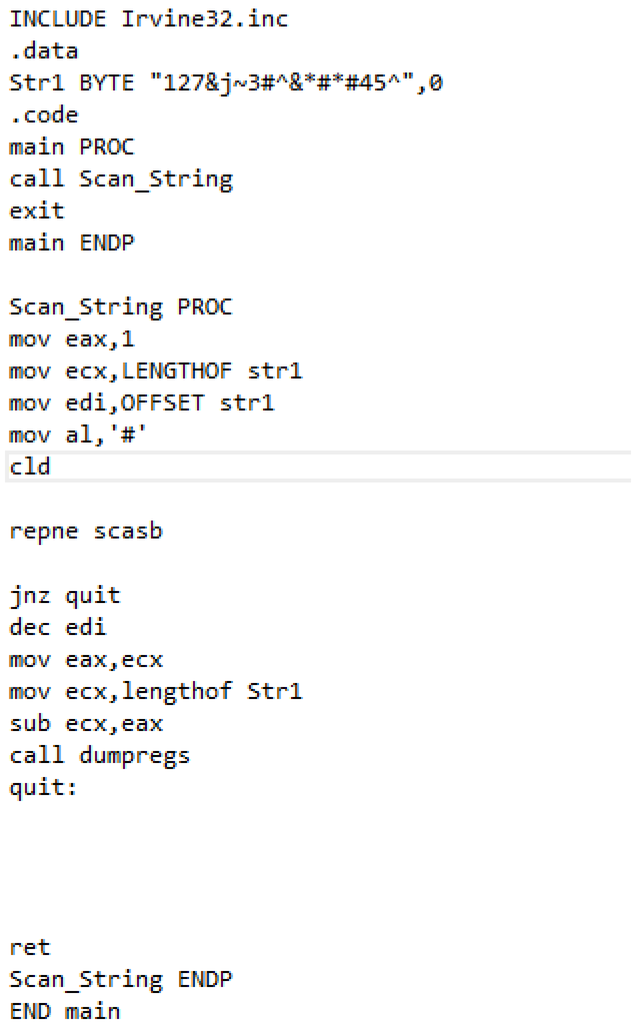
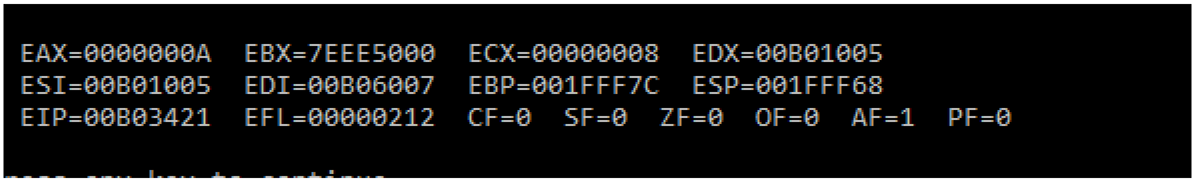
Muzamil p20-0108

Lab #11

Task#1



```
INCLUDE Irvine32.inc
.data
Str1 BYTE "127&j~3#^&*#*#45^",0
.code
main PROC
call Scan_String
exit
main ENDP

Scan_String PROC
mov eax,1
mov ecx,LENGTHOF str1
mov edi,OFFSET str1
mov al,'#'
cld

repne scasb

jnz quit
dec edi
mov eax,ecx
mov ecx,lengthof Str1
sub ecx,eax
call dumpregs
quit:




ret
Scan_String ENDP
END main
```

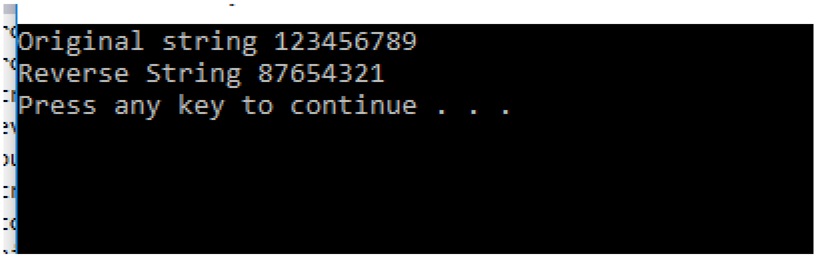


```
EAX=0000000A  EBX=7EEE5000  ECX=00000008  EDX=00B01005
ESI=00B01005  EDI=00B06007  EBP=001FFF7C  ESP=001FFF68
EIP=00B03421  EFL=00000212  CF=0  SF=0  ZF=0  OF=0  AF=1  PF=0

Press any key to continue
```

Task#4

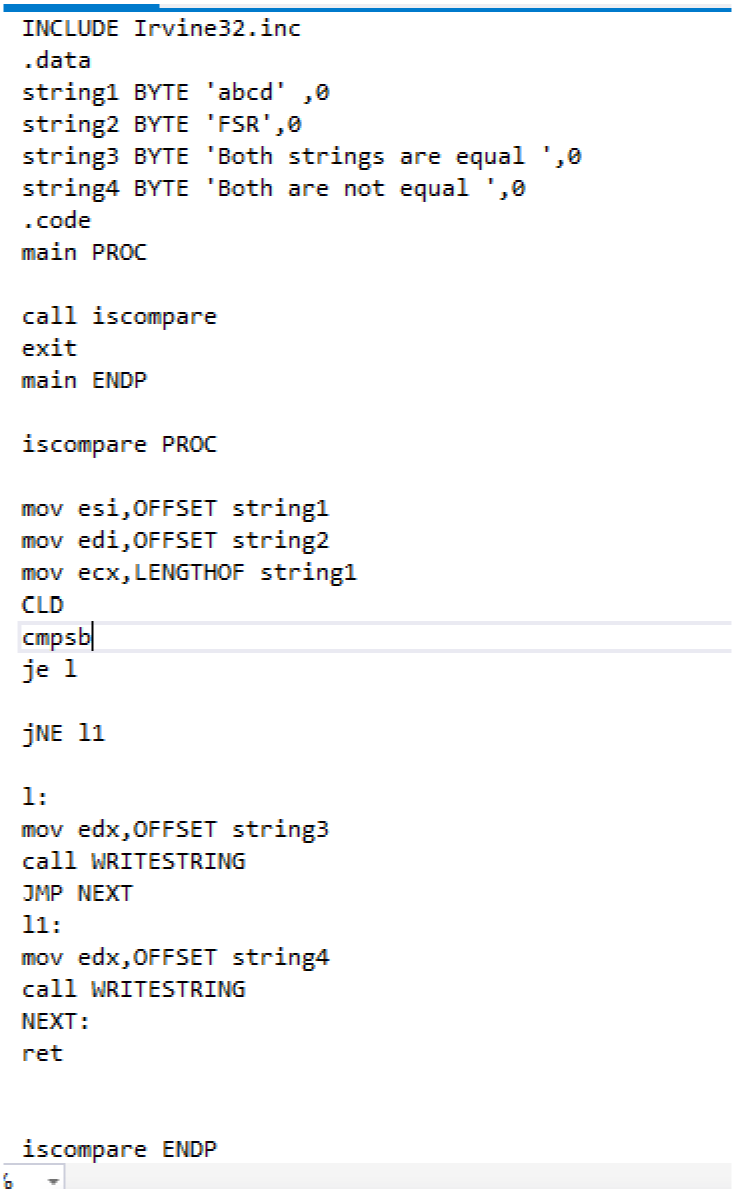
```
INCLUDE Irvine32.inc
.data
prompt1 BYTE "Original string ",0
prompt2 BYTE "Reverse String ",0
str1 BYTE "123456789",0
revtr BYTE LENGTHOF str1
count DWORD 0
str2 BYTE '87654321',0
.code
main PROC
MOV edx,offset prompt1
call writestring
call crlf
mov edx,OFFSET prompt2
call writestring
mov edx,OFFSET str2
call writestring

exit
main ENDP

reverse_string PROC
mov eax,offset str1
mov ecx,LENGTHOF str1
mov ebx,OFFSET revtr
L11:
inc count
loop L11


reverse_string ENDP
END main
```

Output

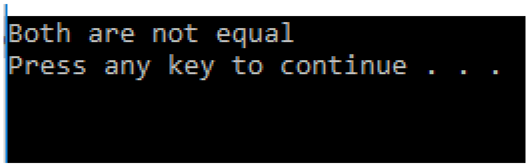


```
Original string 123456789
Reverse String 87654321
Press any key to continue . . .
```

Task#3

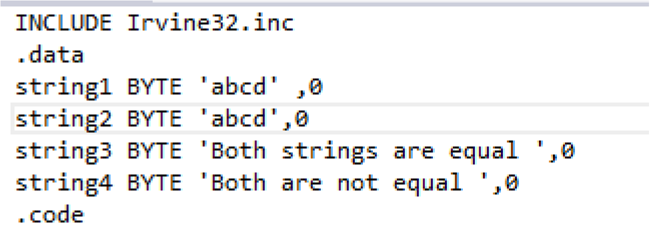


```
INCLUDE Irvine32.inc
.data
string1 BYTE 'abcd' ,0
string2 BYTE 'FSR',0
string3 BYTE 'Both strings are equal ',0
string4 BYTE 'Both are not equal ',0
.code
main PROC

call iscompare
exit
main ENDP

iscompare PROC

mov esi,OFFSET string1
mov edi,OFFSET string2
mov ecx,LENGTHOF string1
CLD
cmpsb
je l

jNE l1

l:
mov edx,OFFSET string3
call WRITESTRING
JMP NEXT
l1:
mov edx,OFFSET string4
call WRITESTRING
NEXT:
ret


iscompare ENDP
```

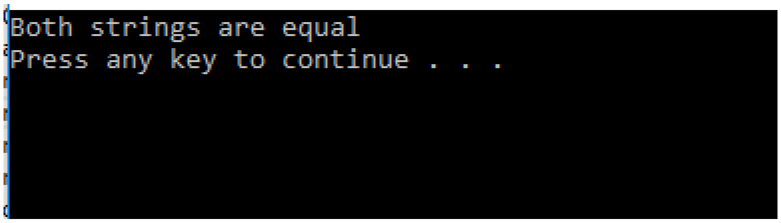


```
Both are not equal
Press any key to continue . . .
```

For same strings

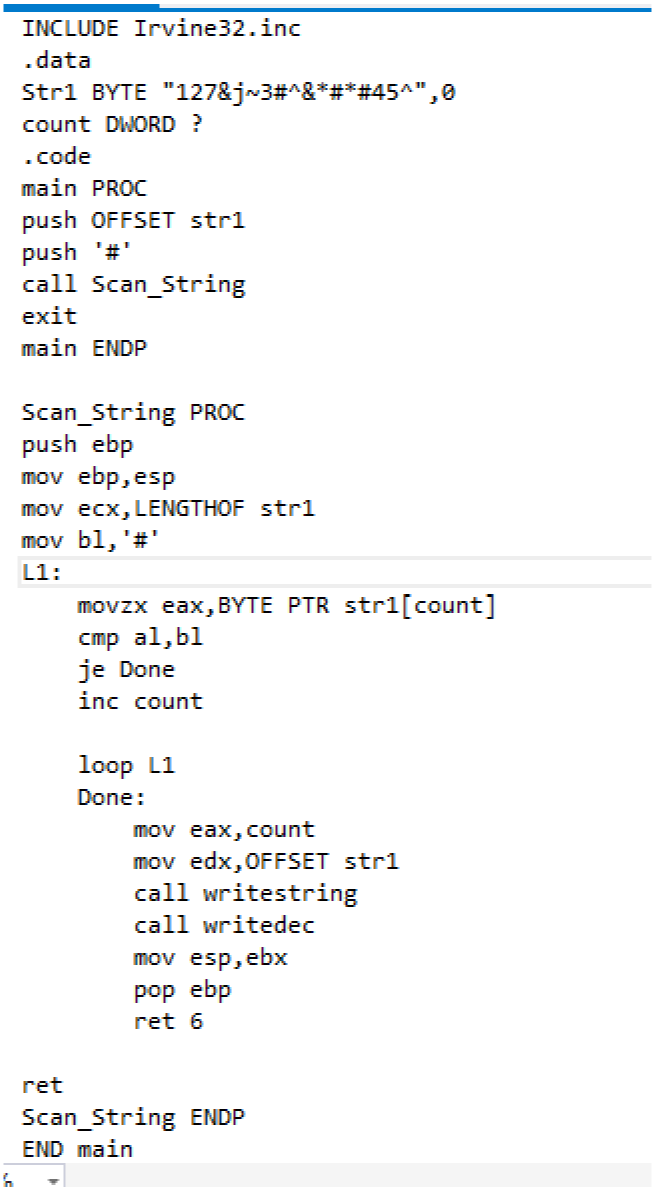


```
INCLUDE Irvine32.inc
.data
string1 BYTE 'abcd' ,0
string2 BYTE 'abcd',0
string3 BYTE 'Both strings are equal ',0
string4 BYTE 'Both are not equal ',0
.code
```

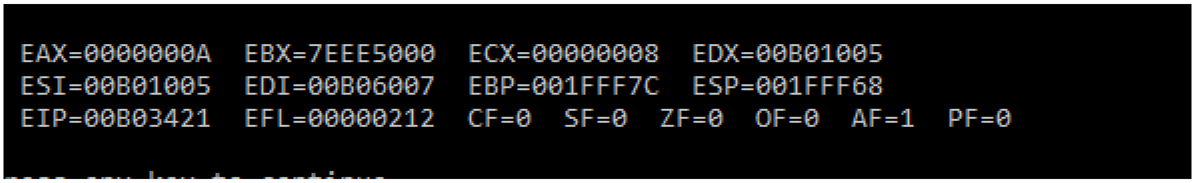


```
Both strings are equal
Press any key to continue . . .
```

Task#2

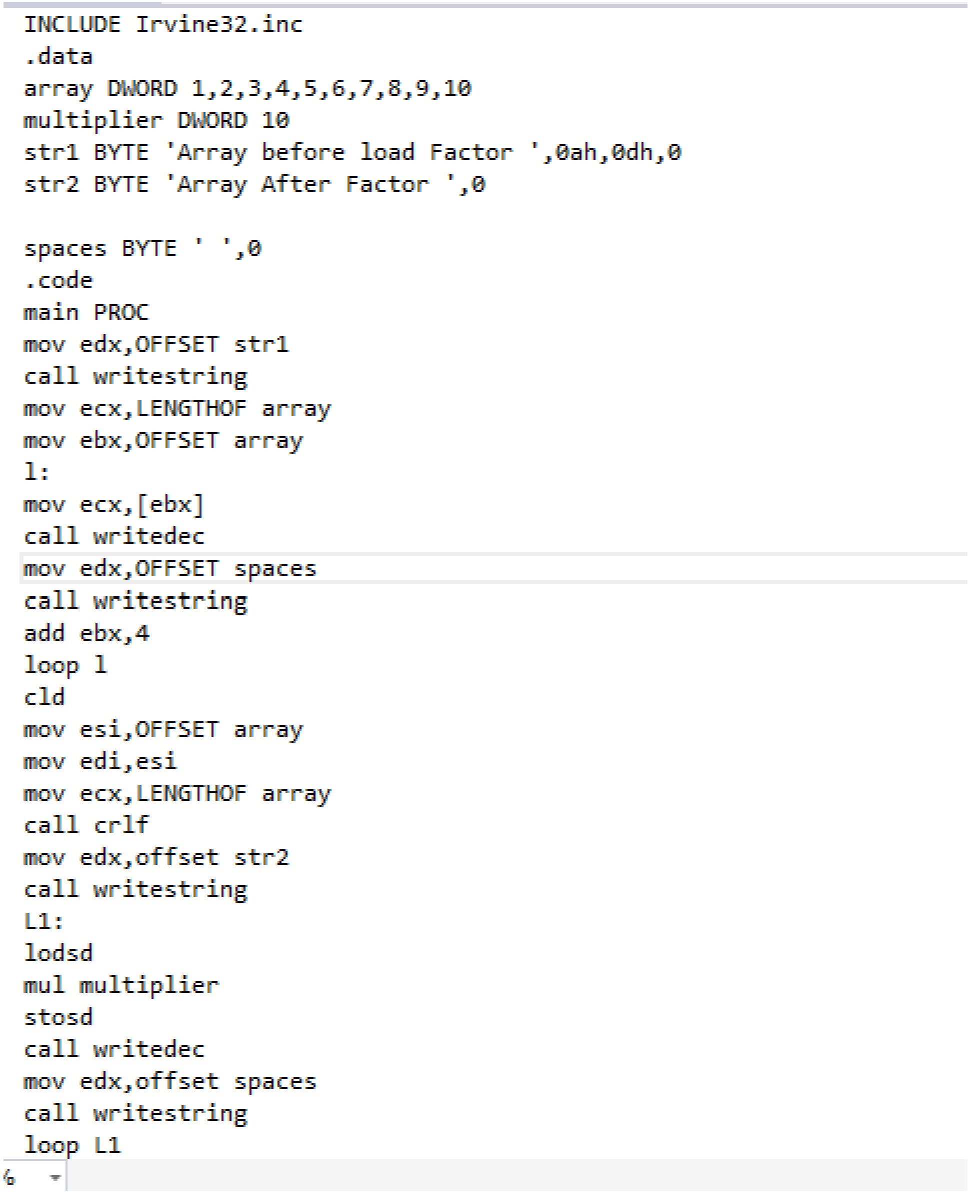


```
INCLUDE Irvine32.inc
.data
Str1 BYTE "127&j~3#^&*#*#45^",0
count DWORD ?
.code
main PROC
push OFFSET str1
push '#'
call Scan_String
exit
main ENDP

Scan_String PROC
push ebp
mov ebp,esp
mov ecx,LENGTHOF str1
mov bl,'#'
L1:
    movzx eax,BYTE PTR str1[count]
    cmp al,bl
    je Done
    inc count

    loop L1
    Done:
        mov eax,count
        mov edx,OFFSET str1
        call writestring
        call writedec
        mov esp,ebx
        pop ebp
        ret 6

ret
Scan_String ENDP
END main
```

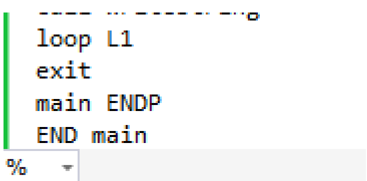


```
EAX=0000000A  EBX=7EEE5000  ECX=00000008  EDX=00B01005
ESI=00B01005  EDI=00B06007  EBP=001FFF7C  ESP=001FFF68
EIP=00B03421  EFL=00000212  CF=0  SF=0  ZF=0  OF=0  AF=1  PF=0

press any key to continue
```

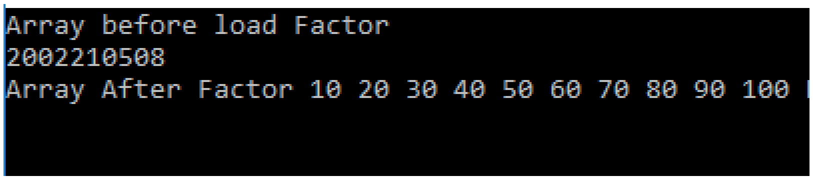
Task#5



```
INCLUDE Irvine32.inc
.data
array DWORD 1,2,3,4,5,6,7,8,9,10
multiplier DWORD 10
str1 BYTE 'Array before load Factor ',0ah,0dh,0
str2 BYTE 'Array After Factor ',0

spaces BYTE ' ',0
.code
main PROC
mov edx,OFFSET str1
call writestring
mov ecx,LENGTHOF array
mov ebx,OFFSET array
l:
mov ecx,[ebx]
call writedec
mov edx,OFFSET spaces
call writestring
add ebx,4
loop l
cld
mov esi,OFFSET array
mov edi,esi
mov ecx,LENGTHOF array
call crlf
mov edx,offset str2
call writestring
L1:
lodsd
mul multiplier
stosd
call writedec
mov edx,offset spaces
call writestring
loop L1
```

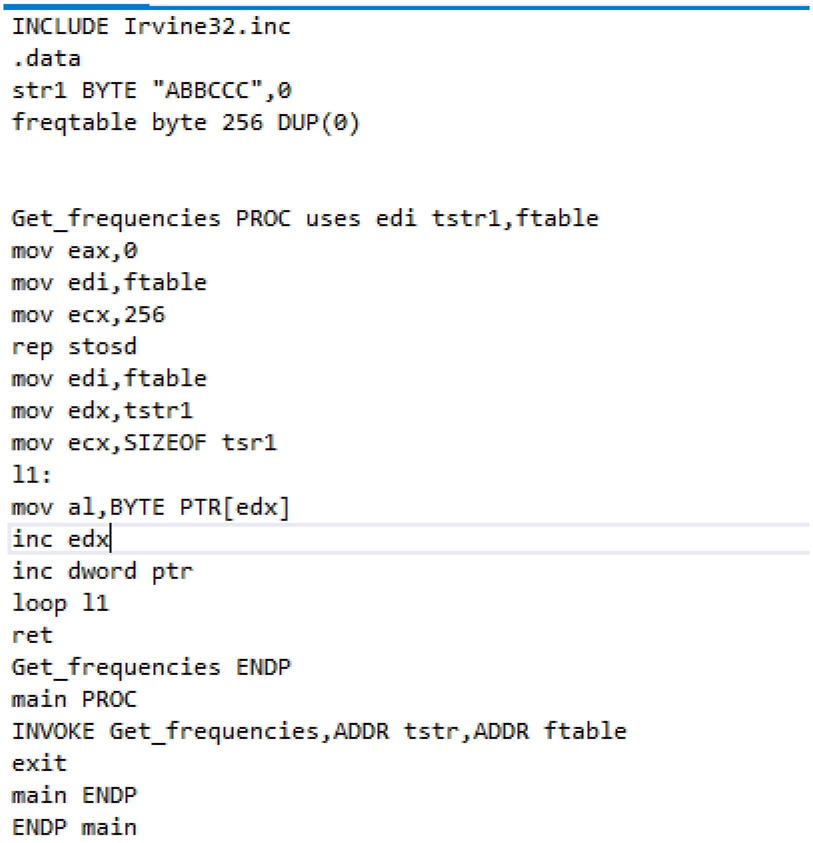


```
loop L1
exit
main ENDP
END main
```



```
Array before load Factor
2002210508
Array After Factor 10 20 30 40 50 60 70 80 90 100
```

Task#6



```
INCLUDE Irvine32.inc
.data
str1 BYTE "ABBCCC",0
freqtable byte 256 DUP(0)


Get_frequencies PROC uses edi tstr1,ftable
mov eax,0
mov edi,ftable
mov ecx,256
rep stosd
mov edi,ftable
mov edx,tstr1
mov ecx,SIZEOF tsr1
l1:
mov al,BYTE PTR[edx]
inc edx
inc dword ptr
loop l1
ret
Get_frequencies ENDP
main PROC
INVOKE Get_frequencies,ADDR tstr,ADDR ftable
exit
main ENDP
ENDP main
```